SOME POINTS AFFECTING WEB PERFORMANCE

Christophe Deleuze
Laboratoire de conception et d'intégration des systèmes (LCIS)
50, rue Barthélémy de Laffemas, BP54 26902 Valence Cedex 09 France

Keywords: Web performance, HTTP, persistent connections, embedded objects, web content.

Abstract: Despite continuously growing network and server capacity, web performance is still often bad. Changes in version 1.1 of the hypertext transfer protocol (HTTP) have brought solutions to some of the performance problems, but daily experience shows that problems remain. In this work, we examine some characteristics of a number of popular web pages and try to get insights about what could be made to enhance overall web performance. We find that most web pages have very large numbers of small objects, something HTTP doesn't handle very well, and that style-sheets were expected to avoid. Moreover, half of the sites we examined do not support HTTP enhancements such as persistent connections and request pipelining. Finally, most sites embed objects from multiple (and often at lot of) different hosts.

1 INTRODUCTION

The Internet is now a production network, used for daily business. The advent of the world wide web and its user friendly browsing capabilities has been the major step in making the Internet mainstream. Because of its success and the way the system and protocol (HTTP – HyperText Transfer Protocol) were designed, major performance issues have quickly appeared. Things were getting bad enough for many people to call it the "world wide wait". Researchers involved in the design of Internet protocols have even called "abysmal" the web protocols (Jacobson, 1995).

Most major protocol issues were solved with version 1.1 of HTTP, defined in (Fielding et al., 1999). Performance was greatly enhanced by the introduction of features such as *persistent connections*, *buffered transmission*, and *request pipelining* (Krishnamurthy et al., 1999). It is not clear, however, that these features are widely supported.

Despite some attempts to propose new modifications to HTTP (W3C, 2001) it is now considered that its performance is good enough, and that the protocol will not change (Mogul, 1999).

Contrary to previous Internet protocols and usages that were essentially symmetric, arguably *peer to peer*,[1] the web is based on an asymmetric client server model, raising scalability issues. Much work has been done on trying to address such issues: they include various content replication and load balancing techniques such as caching proxies, content distribution networks (CDNs), and server farms (Deleuze, 2004). All these systems require a large and costly infrastructure and fail to solve all the problems. Most notably, the problems of consistency maintenance among replicated servers and replication of dynamic content have proven very difficult to solve.

In the end, while more and more people are getting high speed Internet access, web performance is often not as smooth as people would expect it to be. A recent striking example is the congestion of the french government income tax web site for several weeks, forcing the government to postpone the deadline for e-citizens to declare their income (ZDNet, 2005).

In order to try and find out how things could be enhanced, we have examined the main page of about 500 popular web sites, taken from Alexa's "Global Top 500" list (Alexa, 2005). This paper presents preliminary results of this work. We first focus on the content of the pages in Section 2. In Section 3 we examine the support of two HTTP optional features designed to improve performance: *persistent connections* and *request pipelining*. Finally, in Section 4, we

[1]This is obvious for mail and net news, but also true for client server protocols such as FTP, whose traffic patterns were very symmetric.

Deleuze C. (2006).
SOME POINTS AFFECTING WEB PERFORMANCE.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Internet Technology / Web Interface and Applications*, pages 242-245
DOI: 10.5220/0001246102420245

consider the number of hosts that have to be contacted to download all embedded objects for a page. Conclusions and perspectives of future work are given in Section 5.

2 WEB CONTENT

Probably the main reason why web performance isn't improving as network and server capacity grow is that web content is becoming increasingly rich. Figure 1 shows the cumulative distribution function (CDF) of the number of embedded objects per page. We can see that half of the web pages have more than 25 embedded objects, while almost 10 % have more than 100!

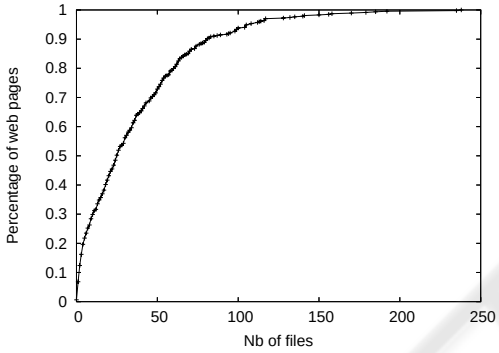

Figure 1: CDF of number of embedded objects.

Table 1 shows the number of documents found and the total byte size for the most popular document types. Pictures clearly constitute the most important type, both when considering number of files and total size. An interesting point to note is how few `png` pictures there are. Globally, 77 % of files are pictures, while they amount for 54 % of bytes.

Table 1: Document types data.

type	number	size (kB)
image/gif	10648	23256
image/jpeg	2903	22839
application/ x-javascript	1281	6632
text/html	868	9416
text/plain	473	413
text/css	452	3309
application/ x-shockwave-flash	207	5836
image/png	86	192
application/ octet-stream	60	1300
image/x-icon	55	144
text/javascript	19	112

Figure 2 shows the mean object and picture size for all the considered pages. Clearly, the mean object size is below a few kilobytes for most of the pages. If we consider only the pictures (that are the vast majority of objects as we already seen) the mean size is even smaller, the median size of pictures for the whole set of pages being 2636 bytes. We must note that an HTTP transaction must be performed for downloading each embedded object, and that each response carry HTTP response headers along with the object. The typical response header size in our downloads was around 300 bytes, i.e. not much less than the objects' sizes themselves.

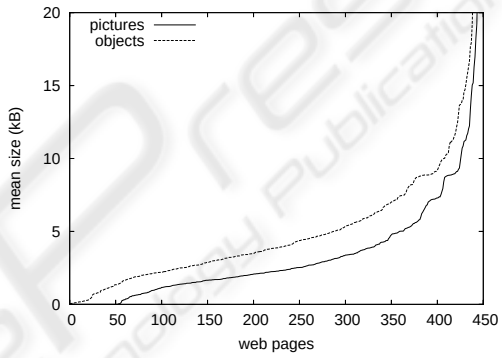

Figure 2: Mean object and picture size.

It is our opinion that such small embedded objects significantly degrade web performance. It was expected that the advent of style-sheets, by providing proper ways for web designers to control page layout, would dramatically increase the mean object size on the web (Nielsen and al., 1997). Apparently, this is still something that can be of some concern.

The second most significant contribution to web pages weight seems to be formed by presentation information such as JavaScript files and style sheets (`text/css`). To evaluate their weight, we divide their size by the size of the whole page, excluding pictures and flash files. Figure 3 shows the numbers (in increasing order) obtained for CSS files, JavaScript files, and both together, for the studied sites. We find that this weight is above 25 % for about one third of the sites, and above 50 % for at least 10 % of them.

To summarize this section, most web pages embed a large number of small pictures, which can significantly affect performance. Other kinds of files such as style sheets and javascripts also have a significant weight, although much lower.

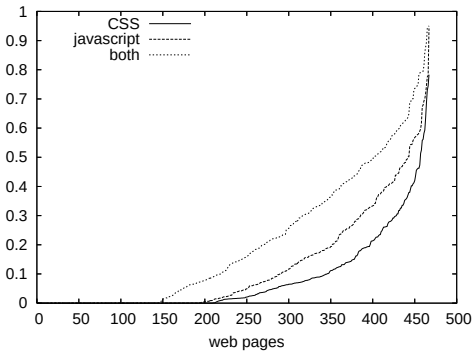

Figure 3: Weigth of CSS and JavaScript files.

3 SUPPORT OF HTTP FEATURES

As we noted in the introduction, a major step in making HTTP more efficient has been the introduction of persistent connections and request pipelining. In the first versions of HTTP, only one transaction could be performed on a given TCP[2] connection. When the web became popular, web pages quickly started to embed lots of small objects, which implied opening and closing many connections for a single page download. Apart from unnecessary traffic in the network and processing at the hosts, this implied significant delays for each transaction. These are due to the time necessary to open each fresh connection, and to the TCP "slow start" congestion control mechanism, that prevents sending data at the full available rate in a just created connection (Padmanabhan and Mogul, 1994).

Persistent connections allow a client to use the same TCP connection for conducting several HTTP transactions to the same server. This saves the time that would be necessary to open a fresh connection, and providing the network isn't congested, the slow start mechanism is impacting only the first few transactions. Following transactions are allowed to send immediately at the full available rate. It can be noted, however, that when transmission restarts on a previously idle TCP connection, the slow start mechanism is re-activated (Allman et al., 1999). This could happen in particular on a slightly overloaded web server.

Even with persistent connections, unnecessary delays may be added if the basic "stop and wait" HTTP transaction model is used. In this case, the client waits for a response for its request before sending the next request. Thus, even if the server answers as quickly as it can, there's still an incompressible delay of one round trip time for each embedded object. *Request pipelining* allows the client to send many requests as in a pipeline. The server then replies to the requests. Note that it can not reorder the requests since HTTP doesn't provide any way to explicitly associate an answer to a request.

We have tested the web sites to try and evaluate to what extend these features are now in use. Our results are shown in Table 2. We found that 244 hosts out of the 494 main hosts for the considered web pages do accept persistent connections. Thus, 50.6 % refuse them. Only 219, i.e. 44.3 % accept the pipelining of requests. These results are particularly significant if we consider the large number of small embedded objects we found in Section 2.

Table 2: Support of persistent connections and pipelining.

feature	number of hosts	
	absolute	%
persistent connections	244	50.6
request pipelining	219	44.3

We're currently analyzing these data. One possible explanation why persistent connections are so unpopular is that server farm systems, such as *web switches* (Deleuze, 2004) are easier to build if persistent connections are avoided. Server farms help in building powerful web servers. Since we have selected popular web sites, many of them may use such systems.

4 MULTIPLE HOSTS

Many web pages have their content scattered on different hosts. Apart from advertisement pictures that are often located on foreign servers, a common situation for a `www.example.com` site is to have all embedded pictures hosted on a second `img.example.com` host. While this may help to balance load on several physical servers, this forces the client to perform several DNS resolutions, and to open TCP connections to several hosts.

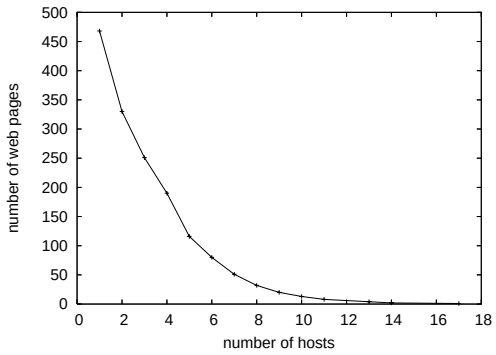

Figure 4: Pages that have at least n hosts.

[2]Transmission Control Protocol, the connection-oriented internet transport protocol HTTP is using.

Figure 4 shows how many web pages have their objects scattered on at least n hosts. One notable point is that more than half of the pages are scattered on at least three hosts, while more than 10 % are on at least seven hosts. We have not tried to estimate what impact the use of multiple hosts can precisely have on the perceived quality, but we're really akin to think that more than a few hosts can't do any good.

Web page content is thus generally spread among several hosts. We now try to find out how well this content is spread. We first define as "the biggest host" for each site the host serving the biggest part (in bytes) of the page. We define its weight as the percentage of the page it owns. Figure 5 shows what the weight of the biggest host is for sites with 2 to 5 hosts.

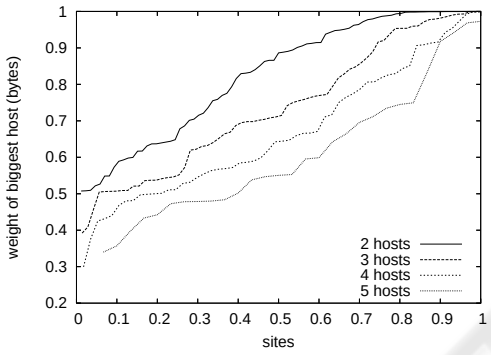

Figure 5: Weight of biggest host (in bytes).

It seems web page content is not well balanced among hosts for most sites. For example, the biggest host owns more than 80 % of the content for 35 % of the three hosts sites.

We don't show them here for lack of space, but roughly similar (slightly better) curves are obtained if we find the biggest host and its weight based on the number of files rather than the number of bytes.

5 CONCLUSION

As a conclusion, we have found that most web pages have lots of embedded objects, and that the mean size of such embedded objects is rather small. This seems to be mainly because of small pictures used extensively in the web pages. Because of the way HTTP has been designed, such small objects can have significant negative effects on the time necessary to download the whole web page. Features such as persistent connections and request pipelining have been introduced in HTTP to mitigate these negative effects, but we found that half of the studied sites do not support them, which is rather interesting.

Finally, our results show that many sites use objects scattered on a large number of hosts. This again can significantly impact performance, mainly by forcing clients to perform many DNS resolutions. Moreover, for most sites the content is not well balanced among hosts.

Another issue we have not considered so far is the effect of server load on TCP persistent connections behavior. As we said, idle connections go in the slow start phase when transmission restarts. Thus, a slight overload on the server, by introducing small delays between responses, may result in much more delays as perceived by the client.

Finally, two points that seem worth investigating are whether correct use of style-sheets can reduce the number of small pictures in the examined web sites, and why persistent connections are so unpopular.

REFERENCES

Alexa (2005). web search. Web site. `www.alexa.com`.

Allman, M., Paxson, V., and Stevens, W. (1999). TCP congestion control. RFC 2581, IETF.

Deleuze, C. (2004). Content networks. *Internet Protocol Journal*, 7(2):2–11. Available at http://www.cisco.com/ipj.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). Hypertext transfer protocol – HTTP/1.1.

Jacobson, V. (1995). How to kill the internet. In *SIGCOMM 95 middleware workshop*.

Krishnamurthy, B., Mogul, J. C., and Kristol, D. M. (1999). Key differences between HTTP/1.0 and HTTP/1.1. In *Proceedings of the WWW-8 Conference*, Toronto.

Mogul, J. C. (1999). What's wrong with HTTP (and why it doesn't matter). In *Proceedings of the 1999 USENIX Annual Technical Conference*, Monterey, CA.

Nielsen, H. F. and al. (1997). Network performance effects of HTTP/1.1, CSS1, and PNG. Technical Report NOTE-pipelining-970624, W3C.

Padmanabhan, V. N. and Mogul, J. (1994). Improving HTTP latency. In *Proceedings of the 2nd World Wide Web Conference*, pages 995–1005, Chicago, IL.

W3C (2001). Final HTTP-NG activity statement. Web page. http://www.w3.org/Protocols/HTTP-NG/Activity.html.

ZDNet (2005). Impots en ligne : Bercy accorde un délai supplémentaire. Web page. http://www.zdnet.fr/actualites/internet/0,39020774,39216771,00.htm.